Pricing Bermudan Option by Binomial Tree

Speaker: Xiao Huan Liu

Course: 74.757.L03

1

Outline

- Introduction
- Problem Definition
- Solution Strategy
- Implementation
- Conclusions and future works

2

Introduction

- A ***Bermudan Option*** is a type of nonstandard American option with early exercise restricted to certain dates during the life of the option. Bermudan Options have an "early exercise" date and expiration date. Before the "early exercise" date, it behaves like a European Option because it can not be exercised. After the "early exercise" date, the option behaves like an American Option because it can be exercised at any time up until expiration.

3

Problem Definition

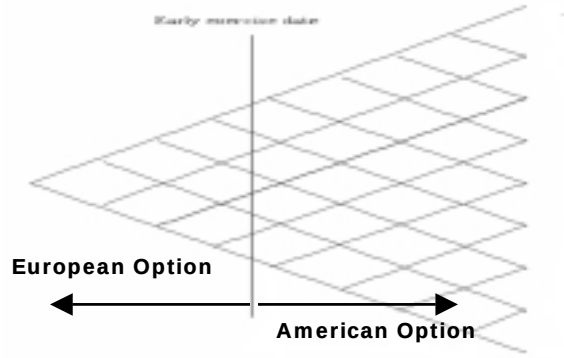
- Price the Bermudan option on a non-dividend-paying stock.
- Approximate the earliest time to exercise the option to gain the holder's expected profit.
- Approximate the optimal time when the holder can gain the best profit.

In my project, I built two models to price the Bermudan option. One model is a standard binomial tree which assumes the volatility is same during the computation. Another model assumes the volatility changes every time interval.

4

Solution Strategy:

- Because of the properties of Bermudan option, we calculate the nodes before the early exercise date like the European option and calculate the nodes after the early exercise date like the American option.

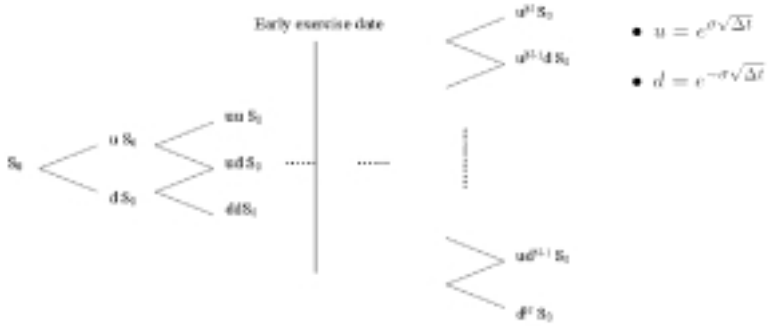


5

Solution Strategy: (continue)

1. We build a binomial tree of stock prices:

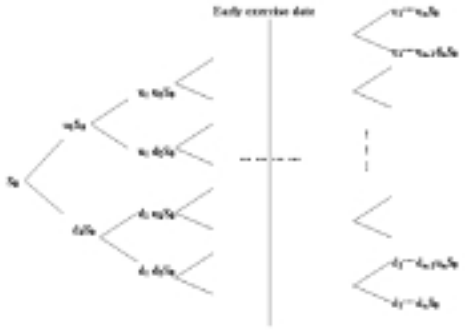
- For the first model in which the volatility is not changed, the diagram looks like the following figure:



6

Solution Strategy: (continue)

- For the second model in which the volatility changes every time interval, the diagram looks like the following figure:

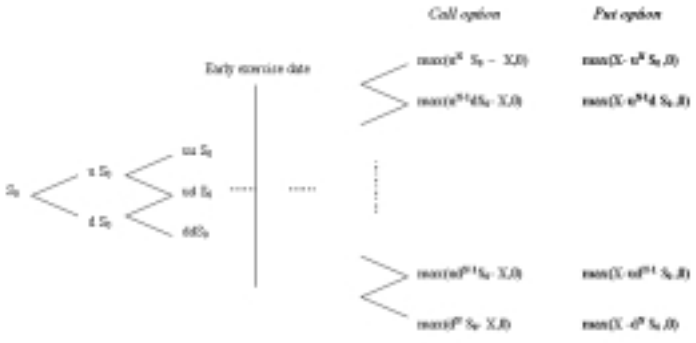


$$\sigma_i = \sigma_{i-1} + \sigma_{i-1} * \Delta t \ (1 \le i \le N)$$
$$u_i = e^{\sigma_i \sqrt{\Delta t}} \ (1 \le i \le N)$$
$$d_i = e^{-\sigma_i \sqrt{\Delta t}} \ (1 \le i \le N)$$

7

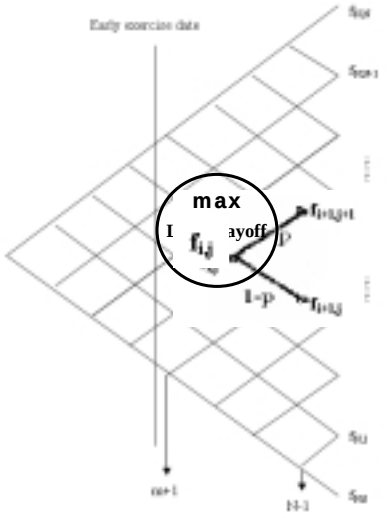
Solution Strategy: (continue)

2. We start compute the option values of the leaf nodes .
   Call option is worth max (S_T-X,0);
   Put option is worth max (X-S_T, 0);
(S_T is the asset price at maturity time, X is the strike price).



8

Solution Strategy: (continue)



3. We assume the nodes from level 0 to level m are before the early exercise date.
Compute backward from level N-1 to level m+1:

- Model I (volatility is same)

$$p = \frac{e^{(r-q)\Delta t} - d}{u - d}$$

$$f'_{i,j} = e^{-r\Delta t}[pf_{i+1,j+1} + (1-p)f_{i+1,j}]$$

- Model II (volatility changes)

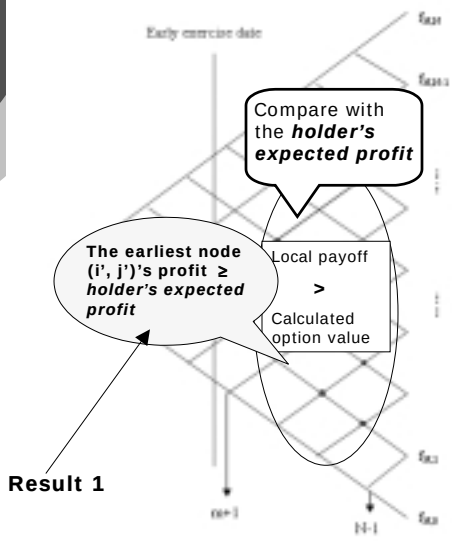
$$p_i = \frac{e^{(r-q)\Delta t} - d_i}{u_i - d_i} \quad (1 \le i \le N)$$

$$f'_{i,j} = e^{-r\Delta t}[p_{i+1}f_{i+1,j+1} + (1-p_{i+1})f_{i+1,j}]$$

Local pay off:
- Put option:
local payoff = max ($X-S_{i,j}$, 0)
- Call Option:
local payoff= max ($S_{i,j}-X$, 0)

$$f_{i,j} = max(localpayoff, f'_{i,j})$$

Solution Strategy: (continue)



4.For the nodes whose local payoff are greater than the calculated option value, it means that when the stock price reaches these nodes, exercising the option is better than waiting.
- Calculate the early exercise profit of these nodes.

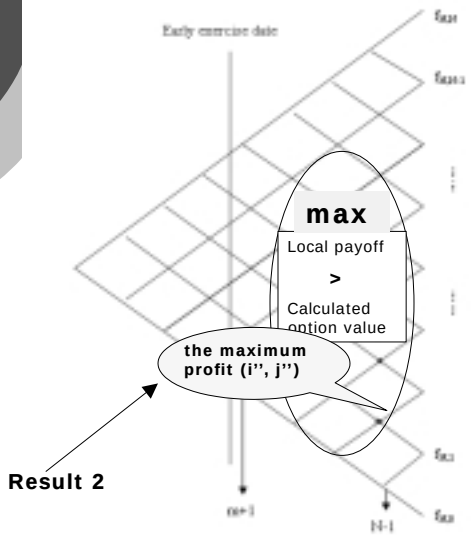
$$profit_{i,j} = \frac{localpayoff - f'_{i,j}}{localpayoff}$$

- compare these profits with holder's expected profit to get the node whose profit is greater than or equal with the holder's expected profit and whose i is the minimum among these nodes.

Result 1:

The earliest time to exercise= i' *Δt

Solution Strategy: (continue)

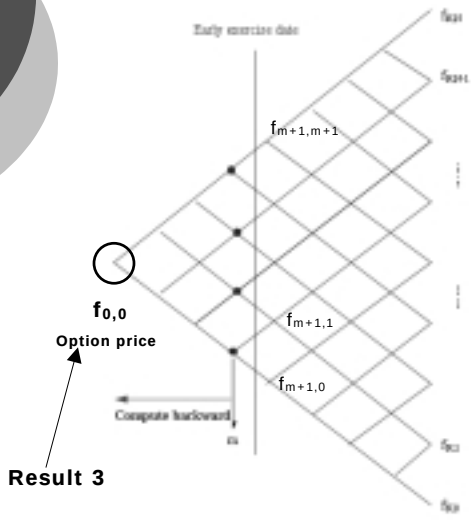


5.Compare the profits of all these nodes to get the maximum one:

Result 2:

The best time to exercise=i'' * Δt

Solution Strategy: (continue)



6. After we calculate the option values on the level m+1, we continue to calculate backward. The nodes from level 0 to level m like the nodes on the binomial model of an European option.(0≤ i ≤ m)

- Model I (volatility is same)

$$f_{i,j} = e^{-r\Delta t}[pf_{i+1,j+1} + (1-p)f_{i+1,j}]$$

- Model II (volatility changes)

$$f_{i,j} = e^{-r\Delta}[p_{i+1}f_{i+1,j+1} + (1-p_{i+1})f_{i+1,j}]$$

Result 3:

the option value of node(0,0) is the option price.

Implementation

Development Tool: Microsoft Visual C++ 6.0

Interface:

Input parameters:

- Current Stock Price: S_0
- Strike Price: X
- Risk-free interest rate: r
- Volatility: σ
- Step Number (N): N
- Expected profit: *holder's expected profit*
- Option type: *call option or put option*
- change volatility during pricing: *whether the volatility is changed during the computation*
- Option start time
- Early Exercise Day
- Maturity Time

Output (results):

- Computation Price: *option value*
- The earliest time to gain your expected profit
- The best possible profit of early exercise

13

Conclusions and future works:

- Bermudan option is a popular kind of option in the real financial world. To simply the issue, my project just considered the Bermudan option on non-dividend-paying stock.
- When the parameter volatility is not be changed during the computation, we build a standard binomial tree model which has N+1 leaves. With the increase of N, the number of leaves increase linearly.
- When the parameter volatility is changed every time interval, the number of leaves is 2^N. So the number of leaves increases exponentially rather than linearly.

14

Conclusion and future works:

- For the second model, the execute time will increase exponentially when N increases, because the size of the dynamic arrays which are used to store the stock prices and option values is 2^N . As a result, when N reaches a certain value, the memory of the computer will be overflowed. I tested my program on my computer. For the second model, when the N is greater than 22, the execute time increase obviously.
- In real world, the Bermudan option is more complicated. Using binomial model to solve a more complicated Bermudan option need further research.

15